WEST

Help | Logout | Interrupt

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers | Preferences

Search Results -

Terms	Documents
l31 and history with list	24

Database: US Patents Full-Text Database
US Pre-Grant Publication Full-Text Database
JPO Abstracts Database
EPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Refine Search: Clear

Search History

Today's Date: 9/24/2001

DB Name	Query	Hit Count	Set Name
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l31 and history with list	24	L34
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l32 and (step same backward or undo)	19	L33
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l30 and (step same forward or redo)	26	L32
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l30 and step same forward or redo	987	L31
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l29 and snapshot	88	L30
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l28 and history	959	L29
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	document same state	15158	L28
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l25 and redo same command	0	L27
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l25 and undo same command	0	L26
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	5235679.pn.	3	L25
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	5602997.pn.	3	L24
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	5754174.pn.	3	L23
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l20 and undo same command	1	L22
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l17 and undo same command	24	L21
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l16 and redo same command	1	L20
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l16 and (redo or step same forward)	10	L19
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l16 and (undo or step same backward)	7	L18
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l16 and undo or step same backward	9968	L17
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	document same state same history	146	L16
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l12 and history same command$	12	L15
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l12 and history same list	20	L14
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l12 and history with list	7	L13
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l11 and display same commands	214	L12
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l9 and (redo or step same forward same command)	591	L11
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l9 and redo or step same forward same command	2042	L10
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	undo or step same backward same command	4574	L9
USPT	5542088.pn.	1	L8
USPT	5890181.pn.	1	L7
USPT	5907410.pn.	1	L6
USPT	5911074.pn.	1	L5
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l3 and history	19	L4
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l2 and display same commands	46	L3
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	l1 and redo same menu	55	L2
USPT,PGPB,JPAB,EPAB,DWPI,TDBD	undo same menu	268	L1

Generate Collection

L34: Entry 2 of 24 File: USPT Jul 3, 2001

US-PAT-NO: 6256032
DOCUMENT-IDENTIFIER: US 6256032 B1

TITLE: Method and apparatus for organizing and processing information using a digital computer

DATE-ISSUED: July 3, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Hugh; Harlan M.	Los Angeles	CA	N/A	N/A

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
TheBrain Technologies Corp.	Santa Monica	CA	N/A	N/A	02

APPL-NO: 9/ 487701
DATE FILED: January 19, 2000

PARENT-CASE:
CROSS-REFERENCE TO RELATED APPLICATIONS This application is a continuation of allowed U.S. patent application Ser. No. 08/892,548, filed Jul. 14, 1997, now U.S. Pat. No. 6,031,537 which is a continuation-in-part of allowed U.S. patent application Ser. No. 08/747,092, filed Nov. 7, 1996, now U.S. Pat. No. 6,037,944.

INT-CL: [7] G06F 3/00
US-CL-ISSUED: 345/357; 345/349
US-CL-CURRENT: 345/854; 345/764
FIELD-OF-SEARCH: 345/326-358

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4974173	November 1990	Stefik et al.	364/521
☐	5220657	June 1993	Bly et al.	395/425
☐	5459831	October 1995	Brewer et al.	395/155
☐	5506937	April 1996	Ford et al.	395/12
☐	5555354	September 1996	Strasnick et al.	395/127
☐	5619632	April 1997	Lamping et al.	395/141
☐	5666503	September 1997	Campanelli et al.	345/356
☐	5786820	July 1998	Robertson	345/357
☐	6031537	February 2000	Hugh	345/357

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTR	US-CL
44 40 598 C1	May 1996	DEX	
0 273 435 A2	July 1988	EPX	
0 325 777 A2	August 1989	EPX	

OTHER PUBLICATIONS

Andres, MacUser, v.10 n1, P(63), "Canvas 3.5", Debeba Software, Jan. 1994.*
Halasz et al., "NoteCards in a Nutshell", 1987, pp. 45-52, ACM.
Carolyn L. Foss, "Tools For Reading And Browsing Hypertext", 1989, pp. 407-418, Information Processing & Management, vol. 25, No. 4.
Olsen et al., "Visualization of a Document Collection: The Vibe System", vol. 29, No. 1, Jan. 1993, pp. 69-81, Information Processing & Management.
Kent, et al., "Creating A Web Analysis and Visualization Environment", vol. 1, No. 28, Dec. 1995, pp. 109-117, Computer Networks and ISDN Systems.
Carr et al., "Open Information Services", vol. 28, No. 11, May 1996, pp. 1027-1036, Computer Networks and ISDN Systems.
Inspiration 5.0 Pro--Review, http://www.ozemail.com.au/.about.caveman/Creative/Software/Inspiration/ index.html, Feb. 6, 1998.
VisiMap, http://www.coco.co.uk.prodvm.html?tag.dbd.ex.cn.3672tcc.snap.sn, Sep. 19, 2000.
Miller et al., "SemNet.RTM. User Guide, Version 1.1", 1994, pp. i-xi, 1-146, SemNet Research Group.

ART-UNIT: 213
PRIMARY-EXAMINER: Sax; Steven
ATTY-AGENT-FIRM: Chou; Chien-Wei (Chris) Oppenheimer Wolff & Donnelly LLP

ABSTRACT:

A method and apparatus for organizing and processing pieces of interrelated information (or "thoughts") using a digital computer is disclosed. The invention employs a graphical user interface to facilitate user interaction with highly flexible, associative "matrices" that enable users conveniently to organize digitally-stored thoughts and their network of interrelationships. Each of the thoughts may be affiliated with one or more application programs, such as a word processing or spreadsheet utility, or an Internet browser. Users are able conveniently to select a current thought along with any applications or content associated with that thought by interacting with the graphical representation. That representation is automatically reoriented about the selected thought, and is revised to reflect only those thoughts having predetermined relations to that current thought. Users can easily modify the matrix by interactively redefining relations between thoughts. Further aspects of the invention include techniques permitting automated generation of thought matrices, delayed loading to facilitate navigation amongst thoughts without undue delay due to bandwidth constraints, and matrix division and linking to allow optimal data structure flexibility. Finally, the present invention is interoperable with computer networks including the internet, and offers an intuitive scalable methodology for the navigation and management of essentially immeasurable information resources and knowledge bases that transcends the limitations inherent in traditional hierarchical approaches.

36 Claims, 25 Drawing figures

DOCUMENT-IDENTIFIER: US 5551055 A
TITLE: System for providing locale dependent user interface for presenting control graphic which has different contents or same contents displayed in a predetermined order

DEPR:
Finally: what are the commands that can operate on this selection? In a word processing program, a command might change the style of a selected range of characters and in a structured graphics program, a command might rotate a graphic object. The subject invention provides a large number of built-in command objects for all of the built-in data types as well as providing generic commands for Cut, Copy, Paste, Starting HyperMedia Links, Completing Links, Navigating Links, Pushing Data on Links, Pulling Data on Links, as well as many user interface commands. The abstract baseclass that represents a command made by the user is responsible for capturing the semantics of a user action, determining if the command can be done, undone, and redone. Command objects are responsible for encapsulating all of the information necessary to undo a command after a command is done. Before a command is done, command objects are very compact representations of a user action. The baseclass is independent of the user interface technique used to create them. Commands are typically created from menus or via direct manipulation by the user (e.g. moving a graphic object) but could be created via a script. This orthogonality with the user interface is very important.

DEPR:
The invention is designed to support multi-level undo. Implementing this feature, however, requires no extra effort on the part of a developer. The system simply remembers all the command objects that are created. As long as the corresponding command object exist, a user can undo a particular change to the data. Because the system takes care of saving the commands and deciding which command to undo or redo, a user does not implement an undo procedure.

DEPR:
A portion of the data encapsulator protocol deals with filing the data into a stream and recreating the data at another place and/or time. The system uses this protocol to implement document saving. By default, a user's data objects are streamed to a file when saved. When the document is opened, the data objects are recreated. The system uses a data management framework to ensure the data written to disk is in a consistent state. Users tend to save a file often so that their data will be preserved on disk if the system crashes. The subject invention does not require this type of saving, because the system keeps all the command objects. The state of the document can be reconstructed by starting from the last disk version of the document and replaying the command objects since that point in time. For reliability, the system automatically logs command objects to the disk as they occur, so that if the system crashes the user would not lose more than the last command.

DEPR:
The invention also supports document versioning. A user can create a draft from the current state of a document. A draft is an immutable "snapshot" of the document at a particular point in time. (One reason to create a draft is to circulate it to other users for comments.) The system automatically takes care of the details involved with creating a new draft.

DEPR:
As mentioned above, a document can be reconstructed by starting with its state at some past time and applying the sequence of command objects performed since that time. This feature allows users to recover their work in the case of a crash, and it can also be used to support real-time collaboration. Command objects operate on selections, which are address-space independent. Therefore, a selection object can be sent to a collaborator over the network and used on a remote machine. The same is true of command objects. A command performed by one collaborator can be sent to the others and performed on their machines as well. If the collaborators start with identical copies of the data, then their copies will be remain "in sync" as they make changes. Creating a selection is done using a command object, so that all collaborators have the same current selection.

DEPR:
AdoptData must be implemented by the derived class to support absorbing or embedding data into the specification's associated representation. If the data is to be absorbed it must be of a type which can be incorporated directly into the receiver's representation. The absorbed data is added to the representation as defined by the specification. It is common for many data types to replace the currently specified data with the newly absorbed data. Any replaced data is returned in a data encapsulator to support Undo. If the data is to be embedded, the encapsulator is incorporated as a black box and added as a child of the representation.

DEPR:
Creating a new class of command involves overriding a number of methods. The most important three methods to override are: HandleDo, HandleUndo and HandleRedo. The HandleDo method is responsible for changing the data encapsulator appropriately based on the type of command that it is and the selection the command is applied to. For example, if the command involves a style change to a range of characters in a word processor, the HandleDo method would call a method (or set of methods) in the data encapsulator to specify a character range and style to change. A more difficult responsibility of the HandleDo method is saving all of the information necessary to "undo" this command later. In the style change example, saving undo information involves recording the old style of the character range. The undo information for most commands is very simple to save. However, some commands, like find and change may involve recording a great deal of information to undo the command at a later time. Finally, the HandleDo method is responsible for issuing change notification describing the changes it made to the data encapsulator.

DEPR:
The HandleUndo method is responsible for reverting a document back to the state it was in before the command was "done." The steps that must be applied are analogous to the steps that were done in the HandleDo method described above. The HandleRedo method is responsible for "redoing" the command after it had been done and undone. Users often toggle between two states of a document comparing a result of a command using the undo/redo combination. Typically, the HandleRedo method is very similar to the HandleDo method except that in the Redo method, the information that was derived the last time can be reused when this command is completed (the information doesn't need to be recalculated since it is guaranteed to be the same).

DEPR:
Command objects capture the semantics of a user action. In fact, a command represents a "work request" that is most often created by a user (using a variety of user interface techniques) but could be created (and applied) in other ways as well. The important concept is that command objects represent the only means for modifying the data contained in a data encapsulator. All changes to the data encapsulator must be processed by a command object if the benefits of infinite undo, save-less model, and other features of the invention are to be realized.

DEPR:
Model based tracking is the best solution for tracking in documents, but it does have the drawbacks that: (1) the model's views must be optimized to provide quick response to change events and (2) the model must be capable of expressing the intermediate track states.

DEPR:
Persistent selections or "anchors" are very similar to selections in that they are specifications of data in a representation. The difference is that anchors must survive editing changes since by definition anchors persist across changes to the data. The implementation of graphics selections described earlier in the document is persistent. The implementation of text selections, however, is not. If a user inserts or deletes text before a selection, then the character offsets must be adjusted. There are a couple of approaches for implementing text anchors. First, the text representation maintains a collection of markers that point within the text, similar to the way styles are maintained. The anchors include an unique id that refers to a marker. When the text is changed, the appropriate markers are updated, but the anchors remain the same. Another approach is to maintain an editing history for the text. The anchor could contain a pair of character positions, as well as a time stamp. Each time the text was edited, the history would be updated to record the change (e.g., 5 characters deleted from position X at time T). When the anchor is used, the system would have to correct its character positions based on editing changes that happened since the last time it was used. At convenient times, the history can be condensed and the anchors permanently updated.

DEPR:
Whenever a user action invokes any command as shown in input block 1270, a user causes a command to be executed. This could be from a menu item, control, or through direct

manipulation of an object. The action causes a document state be modified as shown in function block 1280, and a document sends notification as shown in function block 1290. When a document sends notification, the following steps are executed: 1) any menu item (or other control) connected for the notification sent by the document receives a notification message. This message includes the name of the change as well as a pointer to the object that sent the notification) a menu item then updates its state, and control is passed back to function block 1230 for further processing.

DEPR:
FIG. 13 is an illustration of a display in accordance with the subject invention. The menu item is Edit 1300 and has a number of sub-menu items associated with it. Undo 1310 is an active menu item and can thus be selected to carry out the associated functions. Redo 1320 is inactive and is thus presented in a greyed out fashion and cannot be selected at this time. A checkbox is also shown at 1360 as part of the debugging control panel 1350.

DEPR:
Other sets of controls are designed to work together and should be undone and redone as an atomic operation. This is accomplished by putting a mark on the undo stack when the dialog box or control is started. When finished, either by dismissing the control panel or when the user presses an OK button (as in IIB above), all of the commands executed since the mark was placed on the undo stack are collected together into a single command group. This group can then be undone or redone as a single group.

DEPR:
Control panels containing a CANCEL button (usually accompanied by an OK button, as in IIB above) us a technique similar to that described III B above. A mark is put on the undo stack when the dialog box or control panel is started. If the user presses the CANCEL button, all commands placed on the undo stack since the mark are undone. This technique works regardless of whether the controls affect the data immediately or not.

DEPR:
The detailed logic of the atomic execution is set forth in the flowchart presented in FIG. 14. Processing commences at terminal 1400 where control is immediately passed to function block 1410 where a dialog box is activated. When the dialog box is activated, a mark is placed on the undo stack. The undo stack is a list of all commands the user has executed. When undo is pressed, the command on the top of the stack is undone. If not immediately redone, it is thrown away. Then, at function block 1410, a user manipulation of a control is detected. The manipulation of a control changes the command's data value, as appropriate as set forth in function block 1430, and executes the control. For example, a checkbox toggles the command's fChecked field between 0 and 1. Finally, the command is recorded on the undo stack so it can be subsequently undone as shown in function block 1440.

DEPR:
As a user subsequently manipulates each control in the dialog box, as detected in decision block 1450, then control passes to function block 1430. However, if a user presses OK as detected in decision block 1460, then control passes to function block 1420. Finally, when each control in the dialog box is set to the user's satisfaction, the user presses the OK button. All of the commands executed since the mark was placed on the undo stack in function block 1440 are collected together into a single command group and placed back onto the undo stack as depicted in function block 1470. A command group is a command that collects many commands together. When executed, undone, or redone, the command group executes, undoes, or redoes each command in sequence. The command group is then placed back onto the undo stack where it can be undone or redone as a single atomic operation.

DEPR:
A title is displayed in a window in order to indicate its purpose. For example, the title for a window to edit a document is usually the name of the document. A label object is used to keep track of the rifle. This label is a graphical object containing a graphic or a text string. As the window changes state, the label automatically adjusts its appearance, without requiring the developer to write additional code. Windows can be either active or inactive. Smart Window label processing is flowcharted in FIG. 16 and the detailed logic is explained with reference thereto.

DEPC:
Multi-level Undo

Generate Collection

L33: Entry 9 of 19 File: USPT Aug 27, 1996

US-PAT-NO: 5551055
DOCUMENT-IDENTIFIER: US 5551055 A

TITLE: System for providing locale dependent user interface for presenting control graphic which has different contents or same contents displayed in a predetermined order

DATE-ISSUED: August 27, 1996

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Matheny; John R.	Mountain View	CA	N/A	N/A
White; Christopher	Mountain View	CA	N/A	N/A
Davis; Mark E.	Cupertino	CA	N/A	N/A

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Taligent, Inc.	Cupertino	CA	N/A	N/A	02

APPL-NO: 7/ 996781
DATE FILED: December 23, 1992

INT-CL: [6] G06F 3/00, G06F 3/03, G06F 3/14
US-CL-ISSUED: 395/882; 395/500, 395/700, 395/892, 364/972.1, 364/943, 364/977.1, 364/927.99
US-CL-CURRENT: 710/62; 703/20, 703/26, 710/72
FIELD-OF-SEARCH: 395/500, 395/700, 395/275, 395/164, 395/882, 395/892

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	3658427	April 1972	Decou	356/156
☐	3881605	May 1975	Grossman	214/1CM
☐	4082188	April 1978	Grimmell et al.	209/73
☐	4635208	January 1987	Coleby et al.	364/491
☐	4677576	June 1987	Berlin Jr. et al.	364/522
☐	4686522	August 1987	Hernandez et al.	340/709
☐	4704694	November 1987	Czerniejewski	364/513
☐	4742356	May 1988	Kuipers	342/448
☐	4821220	April 1989	Duisberg	364/578
☐	4831654	May 1989	Dick	381/51
☐	4885717	December 1989	Beck et al.	364/900
☐	4891630	January 1990	Friedman et al.	340/706
☐	4939648	July 1990	O'Neill et al.	364/422
☐	4953080	August 1990	Dysart et al.	364/200
☐	5008810	April 1991	Kessel et al.	364/200
☐	5041992	August 1991	Cunningham et al.	364/518
☐	5050090	September 1991	Golub et al.	364/475
☐	5060276	October 1991	Morris et al.	382/8
☐	5075848	December 1991	Lai et al.	395/425
☐	5083262	January 1992	Haff, Jr.	395/500
☐	5093914	March 1992	Coplien et al.	395/700
☐	5119475	June 1992	Smith et al.	395/156
☐	5125091	June 1992	Staas, Jr. et al.	395/650
☐	5133075	July 1992	Risch	395/800
☐	5136705	August 1992	Stubbs et al.	395/575
☐	5151987	September 1992	Abraham et al.	395/575
☐	5168441	December 1992	Onarheim et al.	364/146
☐	5177685	January 1993	Davis et al.	364/443
☐	5181162	January 1993	Smith et al.	364/419
☐	5309566	May 1994	Larson	395/275
☐	5329446	July 1994	Kugimiya et al.	364/419.04
☐	5375164	December 1994	Jennings	379/88
☐	5386556	January 1995	Hedin et al.	395/600
☐	5390314	February 1995	Swanson	395/500
☐	5416903	May 1995	Malcolm	395/155
☐	5497319	March 1996	Chong et al.	364/419.02

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	[redacted]UBN-DATE	COUNT[redacted]	US-CL
150273	August 1985	EPX	
398646	November 1990	EPX	

ART-UNIT: 237
PRIMARY-EXAMINER: Lee; Thomas C.
ASSISTANT-EXAMINER: Perveen; Rehana
ABSTRACT:

A method and apparatus for updating an application to conform to unique requirements of a specific locale. The update involves language translation, graphic substitution, and interface element reorientation. For example, the text used in labels, titles, and messages depends upon the selected language. Its direction and orientation may affect the placement and orientation of a menu, menubar, title, scrollbar, or toolbar. Similarly, the selection of icons and other graphical symbols may be culturally dependent. Once localized, user interface elements are stored in a disk dictionary. A disk dictionary is an object that, when given a key, returns a value after reading it in from disk. This disk dictionary is managed by an object called an archive. An archive is responsible for putting together the individual user interface elements that make up a particular presentation.

11 Claims, 21 Drawing figures

Your search matched **7** of **717644** documents.
Results are shown **15** to a page, sorted by **publication year** in **descending** order.
You may refine your search by editing the current search expression or entering a new one the text box.
Then click **Search Again**.

((editable)and (graphical)) and(histories)

Search Again

Results:
Journal or Magazine = **JNL** Conference = **CNF** Standard = **STD**

1 **Prototyping distributed multimedia systems using communicating real-time state machines**
Fortino, G.; Nigro, L.
Real-Time Systems, 2000. Euromicro RTS 2000. 12th Euromicro Conference on , 2000
Page(s): 273 -280

[Abstract] [PDF Full-Text (304 KB)] **CNF**

2 **MPEG-Pro, an authoring system for MPEG-4 with temporal constraints and template guided editing**
Boughoufalah, S.; Dufourd, J.-C.; Bouilhaguet, F.
Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference on , Volume: 1 , 2000
Page(s): 175 -178 vol.1

[Abstract] [PDF Full-Text (532 KB)] **CNF**

3 **Visualizing histories for selective undo and redo**
Chii Meng; Yasue, M.; Imamiya, A.; Xiaoyang Mao
Computer Human Interaction, 1998. Proceedings. 3rd Asia Pacific , 1998
Page(s): 459 -464

[Abstract] [PDF Full-Text (80 KB)] **CNF**

4 **Constraint-based layout in visual program design**
Graf, W.H.; Neurohr, S.
Visual Languages, Proceedings., 11th IEEE International Symposium on , 1995
Page(s): 116 -117

[Abstract] [PDF Full-Text (176 KB)] **CNF**

5 **Visual editing of data structures**
Robson, R.; Seminar, K.
Software Maintenance, 1991., Proceedings. Conference on , 1991

Page(s): 2[redacted]-237

[Abstract] [PDF Full-Text (568 KB)] **CNF**

---

6 **MacSpin: dynamic graphics on a desktop computer**
Donoho, A.W.; Donoho, D.L.; Gasko, M.
IEEE Computer Graphics and Applications , Volume: 8 Issue: 4 , July 1988
Page(s): 51 -58

[Abstract] [PDF Full-Text (660 KB)] **JNL**

---

7 **Editable graphical histories**
Kurlander, D.; Feiner, S.
Visual Languages, 1988., IEEE Workshop on , 1988
Page(s): 127 -134

[Abstract] [PDF Full-Text (588 KB)] **CNF**

Home | Log-out | Journals | Conference Proceedings | Standards
Search by Author | Basic Search | Advanced Search | Join IEEE | Establish a Web Account